C AMIGA

Markt&Technik

SOFTWARE EXTRA

Nr. 4

Best.-Nr.: 38725

GRAFIK

Amiga Object Editor

Komfortables Erstellen von Basic-Bobs

Booter

Bootblock-Grafik für individuelle Disketten

The Bobbit

IFF-Bilder werden zu Bobs und Images

IconMake

Leistungsfähiges Programm zum Wandeln von IFF-Brushes in animierte Icons sowie zum automatischen Umwandeln von Icons in ein C-Programm

C= AMIGA

Markt & Technik

SOFTWARE EXTRA

GRAFIK

Eine 3½″-Diskette

Markt&Technik

Markt & Technik Verlag Aktiengesellschaft · Hans-Pinsel-Straße 2 · D-8013 Haar bei München

Die Informationen in diesem Produkt werden ohne Rücksicht auf einen eventuellen Patentschutz veröffentlicht.
Warennamen werden ohne Gewährleistung der freien Verwendbarkeit benutzt.
Bei der Zusammenstellung von Texten und Abbildungen wurde mit größter Sorgfalt vorgegangen.
Trotzdem können Fehler nicht vollständig ausgeschlossen werden.
Verlag, Herausgeber und Autoren können für fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen.
Für Verbesserungsvorschläge und Hinweise auf Fehler sind Verlag und Herausgeber dankbar.

Alle Rechte vorbehalten, auch die der fotomechanischen Wiedergabe
und der Speicherung in elektronischen Medien.

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
91 90 89 88

Bestell-Nr. 38725

© 1988 by Markt&Technik Verlag Aktiengesellschaft,
Hans-Pinsel-Straße 2, D-8013 Haar bei München/West-Germany
Alle Rechte vorbehalten
Einbandgestaltung: Grafikdesign Heinz Rauner
Druck: Bosch-Druck, Landshut
Printed in Germany

Inhaltsverzeichnis

1	**Einleitung**	5
2	**Amiga Object Editor**	7
2.1	Bedienung des Programms	8
2.2	Maximale Anzahl von Objekten	9
2.3	Bildschirmanordnung	9
2.4	Zeichnen von Objekten	9
2.5	Verschieben von Objekten	9
2.6	Laden und Speichern von Objekten	10
2.6.1	Laden	10
2.6.2	Speichern	10
2.7	Löschen von Objekten	11
2.8	Spiegeln von Objekten	11
2.9	Drehen von Objekten	11
2.10	Kopieren von Objekten	11
2.11	Die Infoseite	12
2.12	Animation von Objekten	12
2.13	Umschalten zwischen den Objekten	12
2.14	Farben ändern	12
2.15	Umschalten auf den Workbench-Bildschirm	13
2.16	Beenden des Programms	13
2.17	Format der *AOE*-Datei	13

3 Booter 14

4 The Bobbit 16

4.1 Das Menü 16
4.2 Grafikfunktionen 17

5 IconMake 18

5.1 Der Aufbau der Info-Dateien 18
5.2 IconMake 31
5.2.1 Das »*Project*«-Menü 31
5.2.2 Das »*Picture 1*«-Menü 32
5.2.3 Das »*Picture 2*«-Menü 33
5.2.4 Das »*Icon Type*«-Menü 33
5.2.5 Das »*Other*«-Menü 33
5.2.6 Der Filename-Requester 37
5.3 Icon_to_C 39

1 Einleitung

Die Stärke des Amiga liegt zweifelsohne in der Fähigkeit, vielfarbige und schnelle Grafik darzustellen. Bei der Erstellung von Bildern kann der Anwender auf viele außergewöhnliche Programme zurückgreifen, die es einfach machen, verblüffende Ergebnisse zu erzielen. Schwieriger wird es schon, wenn man bewegte Grafik in eigene Programme einflechten möchte. Man benötigt dafür recht tiefgehende Programmierkenntnisse.
Um auch dem unerfahrenen Amigafreund die Möglichkeit zu bieten, seinen Computerwerken einen professionellen Charakter zu verleihen, wurde diese Grafikdiskette herausgegeben. Sie enthält als erstes den *»Amiga Object Editor«*, der es erlaubt, Basic-Bobs entweder im Editor selbst zu kreieren oder sie aus IFF-Brushes zu konvertieren. Das Programm enthält komfortable Möglichkeiten, Lage und Position eines Bobs zu verändern. Da mehrere Bobs gleichzeitig editiert werden können, ist es möglich, sie als Animation mit regelbarer Geschwindigkeit abspielen zu lassen.
»The Bobbit« ist ebenfalls ein Programmier-Hilfsprogramm. Es kann IFF-Bilder in Bobs oder *»Images«* wandeln und als C-, Assembler- oder Object-Code speichern. In das Programm sind einige Malfunktionen eingebunden, so daß auch diejenigen, die nicht zu den glücklichen DPaint- oder anderen Malprogramm-Besitzern zählen, mit »The Bobbit« arbeiten können.
»IconMake« beschäftigt sich, wie der Name schon sagt, mit den bedienerfreundlichen Icons der Workbench. Der Autor führt zunächst detailliert in den Aufbau einer Icon-Datei ein, bevor er sich der Bedienung des Programms zuwendet. IFF-Bilder können mit IconMake in Icons verwandelt werden, und durch ein alternatives Icon entsteht der Eindruck einer Animation beim Anklicken. Als Bonbon wird *»Icon_to_C«* mitgeliefert, das ein Icon in Daten umwandelt, so daß es in ein C-Programm integriert werden kann.
»Booter« ist das vierte Programm in dieser Sammlung. Es verleiht jeder Diskette eine persönliche Note, indem es IFF-Bilder mittels *»Trackdisk-Device«* in den Bootblock schreibt, die rasend schnell beim Einlegen der Diskette geladen und angezeigt werden. Ideen für ein solches Startbild sind keine Grenzen gesetzt.

2 Amiga Object Editor

Der *»Amiga Object Editor«* ist ein Programm zur Erstellung von grafischen Objekten (Bobs). Diese können später in eigenen Programmen, zum Beispiel in Spielen, eingesetzt werden. Hierzu kennt der *AOE* zwei verschiedene Speicherformate:

1. Das *AOE*-Format, bei dem mehrere Objekte in einer Datei mit den Farben gespeichert werden.
2. Das normale *»Amiga-Basic«*-Format für Bobs.

2.1 Bedienung des Programms

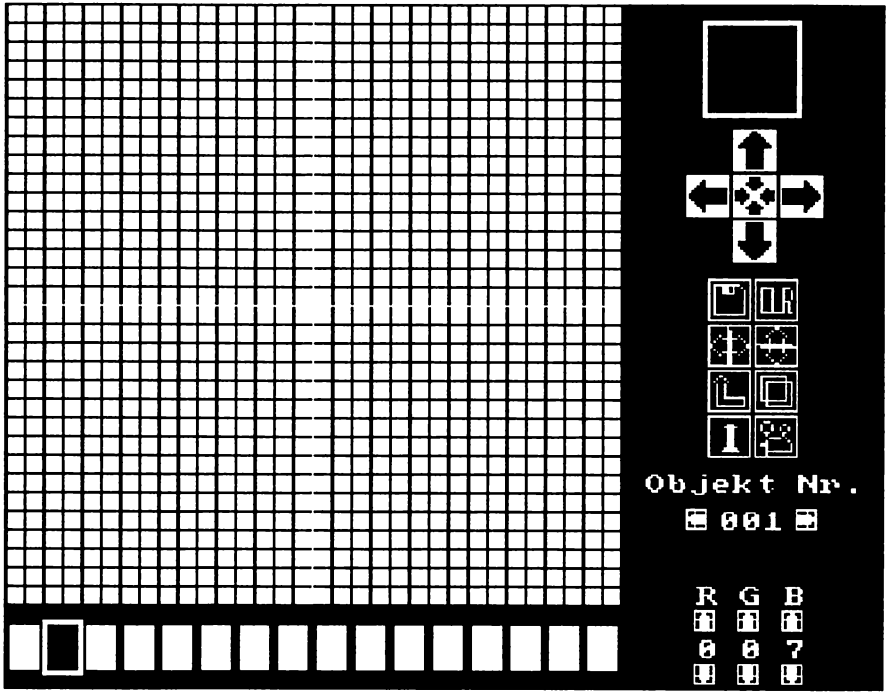


Bild 1. Der Amiga Object Editor

Der *AOE* kann von der Workbench und dem *CLI* gestartet werden. Befinden sich auf der Diskette schon Objektdateien des *AOE*, so können Sie den *AOE* auch durch einen Doppelklick auf das Icon der Objektdatei starten.

2.2 Maximale Anzahl von Objekten

Nach dem Bildschirmaufbau des *AOE* erscheint in der Mitte des Bildschirms ein kleines Fenster, in welchem Sie die maximale Anzahl der zu bearbeitenden Objekte eingeben. Die Anzahl der Objekte ist nur durch das Chip-RAM begrenzt.

2.3 Bildschirmanordnung

Auf der linken Seite sehen Sie das Eingabefenster für die Objekte. Darunter befinden sich die Farb-Gadgets zur Auswahl einer Farbe. Auf der rechten Seite des Bildschirms sehen Sie das Fenster mit dem Objekt in Originalgröße und die Gadgets zur Auswahl der verschiedenen Funktionen.

2.4 Zeichnen von Objekten

Das Zeichnen eines Objekts geschieht vollständig mit der Maus. Um einen Punkt zu setzen, suchen Sie sich zuerst eine der 16 Farben aus. Durch Druck auf den linken Mausknopf wird der Punkt in der aktuellen Farbe gesetzt. Zusätzlich kann durch einen Druck auf die rechte Maustaste ein Punkt gelöscht werden, ohne erst die Hintergrundfarbe auszuwählen.

2.5 Verschieben von Objekten

In der oberen, rechten Hälfte des Bildschirms befinden sich fünf sternförmig angeordnete Gadgets. Mit den äußeren vier Gadgets läßt sich das Objekt in der entsprechenden Pfeilrichtung verschieben, wobei die hinausgeschobenen Punkte auf der gegenüberliegenden Seite wieder erscheinen.

Wenn Sie das mittlere Gadget anklicken, so drehen sich die Pfeile um und zeigen auf die Mitte. In diesem Modus wird das Objekt gestaucht. Das bedeutet, daß durch Anklicken eines Pfeils eine Hälfte des Objekts in Richtung Mittelachse geschoben wird und die frei werdenden Punkte gelöscht werden. Durch nochmaliges Anklicken des mittleren Gadgets drehen sich die Pfeile wieder in die Normalstellung.

2.6 Laden und Speichern von Objekten

Auf der rechten Seite des Bildschirms befindet sich ein Gadget, auf dem eine Diskette abgebildet ist. Durch Anklicken des Gadgets gelangen Sie ins Diskettenmenü. Mit »*Abbruch*« können Sie das Menü vorzeitig verlassen.

2.6.1 Laden

Das Programm fordert Sie auf, die Nummer des Objekts einzugeben, ab welchem die Objekte in den Speicher geladen werden sollen. Nach Eingabe des Dateinamens lädt der *AOE* die Objektdatei. Er erkennt dabei selbständig, um welchen Dateityp es sich handelt. Es gibt folgende Möglichkeiten:

1. *AOE*-Format → Standardformat des *AOE*
2. *IFF*-Brush → *Brush* in der Größe 32 x 32 mit 16 Farben
3. Basic-*Bob* → Basic-Format in der Größe 32 x 32 mit 16 Farben

2.6.2 Speichern

Sie werden vom Programm aufgefordert, die Nummer des ersten und des letzten zu speichernden Objekts einzugeben. Nach Eingabe des Dateinamens speichert der *AOE* die Objekte im *AOE*-Format.

Speichern von Basic-Objekten:
Beim Basic-Format wird immer nur ein Objekt gespeichert. Wenn Sie ein Basic-Objekt speichern möchten, müssen Sie als erstes zwei Buchstaben »*BB*« eingeben. Die beiden Buchstaben werden nicht gespeichert.

Beispiel:
Dateiname = dfl:MeinObjekt:
Sie müssen »*Dateiname = BBdfl:MeinObjekt*« eingeben, um das Objekt im Basic-Format zu speichern.

2.7 Löschen von Objekten

Durch Anklicken des »*CLR*«-Gadgets wird das zur Zeit im Editor befindliche Objekt gelöscht.

2.8 Spiegeln von Objekten

Unterhalb des »*Disketten*«- und des »*CLR*«-Gadgets befinden sich die zwei Gadgets zum horizontalen und vertikalen Spiegeln von Objekten. Durch einfaches Anklicken mit der Maus wird das aktuelle Objekt gespiegelt.

2.9 Drehen von Objekten

Der *AOE* erlaubt es, Objekte in 90°-Schritten zu drehen. Auf dem dazugehörigen Gadget befindet sich ein Pfeil, der um die Ecke zeigt.

2.10 Kopieren von Objekten

Rechts neben dem Gadget zum Drehen um 90° befindet sich das Kopier-Gadget. Nach dem Anklicken erscheint ein Fenster, in das Sie das Objekt eingeben, ab welchem Sie kopieren möchten.
Im nächsten Gadget geben Sie die Nummer des Objekts an, zu dem Sie die Objekte kopieren möchten. Als dritten und letzten Parameter müssen Sie noch die Anzahl der zu kopierenden Objekte eingeben.

2.11 Die Infoseite

Auf der Infoseite sind der Name des Programmautors und ein Copyright-Hinweis vermerkt.

2.12 Animation von Objekten

Bitte klicken Sie hierzu das Gadget mit dem Filmprojektor an. In dem nun folgenden Fenster geben Sie die Nummer des ersten und des letzten zu animierenden Objekts ein. Sofort sehen Sie ein Fenster, in dem die Animation abläuft. Mit den zwei kleinen Pfeilen können Sie die Geschwindigkeit der Animation beeinflussen. Die Zahl gibt die Wartezeit in Fünfzigstelsekunden an.

Beispiel:
Geschwindigkeit = 5
→ Ablaufgeschwindigkeit = $^5/_{50}$ Sekunden = $^1/_{10}$ Sekunden
→ Es wird jede Zehntelsekunde ein neues Objekt dargestellt

2.13 Umschalten zwischen den Objekten

Unter dem Text »*Objekt Nr.*« befinden sich zwei kleine Pfeile, in deren Mitte eine Zahl steht. Durch Anklicken der Pfeile kann um ein Objekt nach oben oder unten geschaltet werden. Wenn Sie die Zahl direkt anklicken, so erscheint ein Fenster, in dem Sie die Nummer des Objekts direkt eingeben können.

2.14 Farben ändern

In der unteren rechten Ecke befinden sich sechs kleine Pfeile, mit deren Hilfe sich die *RGB*-Anteile (Rot, Grün, Blau) einer Farbe verändern lassen.

2.15 Umschalten auf den Workbench-Bildschirm

Mit der Tastenkombination *linke Amigataste* und <*N*> kann auf den Workbench-Bildschirm umgeschaltet werden.
Mit der Tastenkombination *linke Amigataste* und <*M*> kommt man wieder auf den Bildschirm des *AOE* zurück.

2.16 Beenden des Programms

Um das Programm zu beenden, drücken Sie bitte die rechte Maustaste in der rechten oberen Ecke des Bildschirms.

2.17 Format der *AOE*-Datei

Adresse	Wert	Bedeutung
`$00000000.l`	»*AOE*«	Kennung einer *AOE*-Datei
`$00000004.w`	$????	Anzahl der Objekte in der Datei
`$00000006.w`	$????	Es folgen 16 Wörter mit den Farbwerten
		Beispiel $000 = schwarz
		$fff = weiß
		$00f = blau
$00000026	$????	Es folgen nun die Objektdaten:
		Anzahl der Objekte * 640 Byte
1. Objekt		1. Plane (128 Byte)
		2. Plane (128 Byte)
		3. Plane (128 Byte)
		4. Plane (128 Byte)
		5. Maske (128 Byte)
2. Objekt		1. Plane usw.

Die Maske ergibt sich durch eine »*OR*«-Verknüpfung der Planes und gibt an, welche Punkte des Objekts gesetzt sind.

3 Booter

»Booter« ist ein Programm zum Erstellen eines Grafikvorspanns im Bootblock einer Diskette. Dieser Vorspann erscheint danach unmittelbar nach Einlegen der Diskette ins interne Laufwerk. Hierzu wird ein mit DPaint gezeichnetes Bild auf der Diskette abgelegt und beim Booten dargestellt. Dieses Verfahren ermöglicht es auch solchen Leuten, die mit der Programmiererei »nichts am Hut haben«, ihre Disketten mit einem eigenen Vorspann zu versehen.

Bild 2. Disketten erhalten mit Booter eine persönliche Note

Zum Erzeugen eines solchen Vorspanns sind zwei Programme nötig: der *»Booter«*, ein Ladeprogramm, mit dem das Bild und das Vorspannprogramm auf die Diskette geschrieben werden, und *»BootProg«*, das eigentliche Vorspannprogramm im Bootblock. Beide Programme sind für den *MCC*-Makroassembler geschrieben.
Der *»Booter«* legt das Bild übrigens nicht als normales File auf der Disk ab, sondern schreibt es mittels *»Trackdisk-device«* direkt auf die Diskette. Das bringt den Vorteil einer kürzeren Ladezeit; außerdem erscheint das Bild nicht ständig als überflüssiges Anhängsel im Directory.
Die Grafik erstellt man am besten mit DPaint. Um möglichst wenig Platz auf der Disk zu verbrauchen, ist die Auflösung auf zwei Farben, eine Breite von genau 320 Punkten und eine Höhe von höchstens 100 Punkten beschränkt. Man muß das Bild also als Pinsel *(Brush)* speichern. Um die Breite und Höhe richtig zu treffen, sollte man den Koordinatenzähler von DPaint einschalten.

Der Booter wird gestartet mit der Eingabe

Booter [Laufwerk] [Bild]

Dabei ist

[Laufwerk] = *df0:* bis *df3:*

und

[Bild] = der Name (ggf. mit Pfadangabe) des DPaint-Brushes

Das Programm holt sich nun das Bild von der Diskette und bereitet es entsprechend auf. Danach fordert es den Benutzer auf, die zu beschreibende Disk ins gewünschte Laufwerk einzulegen und dann die linke Maustaste zu drücken.
Nach einem Druck auf diese Maustaste prüft der Booter zunächst, ob wirklich eine Disk im angesprochenen Laufwerk liegt und ob diese Disk nicht schreibgeschützt ist. Andernfalls wird ein entsprechender Requester geöffnet, um den Benutzer auf seinen Fehler aufmerksam zu machen.
Anschließend sucht das Programm acht freie, direkt aufeinander folgende Blöcke für das Bild, kennzeichnet sie als belegt, schreibt das Bild in diese Blöcke und das Vorspann-Programm in den Bootblock.
Wenn die Diskette schon den *»Booter«*-Vorspann trägt, werden die bereits reservierten Blöcke des vorherigen Bildes wiederverwendet, um nicht erneut Platz auf der Disk zu verbrauchen.
Es kann vorkommen, daß der Booter *»Disk voll«* meldet, obwohl das CLI-Kommando *»Info«* noch mehr als acht freie Blöcke anzeigt.
Diese Blöcke sind dann nicht aufeinander folgend. In einem solchen Fall hilft es meistens, ein Programm (zum Beispiel ein CLI-Kommando) von der Disk zu löschen, den Booter nochmals anzuwenden und dann das File zurückzukopieren (falls man es unbedingt braucht).
Wenn man nun die so behandelte Disk nach einem Reset ins interne Laufwerk legt, sollte man sein Bild vor einem farbig scrollenden Hintergrund auf- und abspringen sehen.

4 The Bobbit

»The Bobbit« ist als Programmierer-Utility gedacht. Es dient dazu, *IFF*-Brushes (zum Beispiel aus DPaint) in *»Bobs«* und *»Images«* zu konvertieren, wobei man den Code in verschiedenen Formaten speichern kann, um ihn dann in Sprachen wie Basic, C und Assembler zu implementieren. Für die Amiga-Benutzer, die sich nicht zu den glücklichen DPaint-Besitzern zählen können, haben wir in Bobbit ein einfaches Malprogramm mit diversen Grafikfunktionen, wie Drehen, Spiegeln, Halbieren, Invertieren etc., eingebunden.

4.1 Das Menü

Um die Optionen des Menüs nutzen zu können, müssen Sie zuerst *»MENU«* im *»Mode«*-Requester (das sind die Kästchen mit *»MENU«* und *»DRAW«*) anklicken. Wenn Sie nun mit dem Mauszeiger in die linke obere Ecke fahren und die rechte Maustaste drücken, können Sie die gewünschte Option anwählen:

LOAD BBT	lädt ein File, das zuvor mit *»SAVE BBT«* gespeichert wurde.
LOAD IFF	lädt ein *IFF*-Brush mit einer Größe von maximal 32*32 Pixel (nur Lo-Res).
SAVE BBT	speichert die angezeigte Grafik im Bobbit-Format.
SAVE BAS	speichert die angezeigte Grafik als Basic-Bob (die *»OBJECT.SHAPE«* des Amiga-Basic verwenden!).
SAVE C-S	speichert die angezeigte Grafik als *C*-Code.
SAVE A-S	speichert die angezeigte Grafik als Assembler-Code.
SAVE SRC	speichert die angezeigte Grafik als Object-Code.
DELETE	löscht das im Text-Gadget angegebene File.
QUIT	beendet das Programm und kehrt ins CLI zurück.

All diese Optionen benötigen selbstverständlich einen Filenamen, den Sie zuvor im Text-Gadget (dem länglichen Kästchen auf der rechten Bildschirmhälfte) eintippen müssen. Einfach anklicken, eingeben und <*Return*> drücken.

Drawing Wenn Sie eigene Objekte mit Bobbit erstellen wollen, müssen Sie zuerst »*DRAW*« im »*Mode*«-Requester anklicken. Am rechten Bildschirmrand finden Sie die Colorpalette. Hier können Sie die gewünschte Malfarbe per Druck auf die linke Maustaste auswählen. Die Palette läßt sich mit den *RGB*-Reglern einfach editieren. Im »*Magnify*«-Window können Sie die angezeigte Grafik bearbeiten. Mit der linken Maustaste setzen Sie Punkte in der angezeigten Farbe, mit der rechten in Farbe 0.

4.2 Grafikfunktionen

»*The Bobbit*« bietet diverse Optionen zum Editieren der angezeigten Grafik.

CLR löscht das »*Magnify*«-Window mit Farbe 0.
INV invertiert die Grafik und macht voneinander nicht zu unterscheidende Punkte sichtbar.
TURN LEFT dreht die Grafik um 90° nach links.
TURN RIGHT dreht die Grafik um 90° nach rechts.
HALF X halbiert die Grafik nach der X-Achse.
HALF Y halbiert die Grafik nach der Y-Achse.
FLIP X spiegelt die angezeigte Grafik an der X-Achse.
FLIP Y spiegelt die Grafik an der Y-Achse.
SHIFT UP, DOWN, RIGHT, LEFT verschiebt die Grafik nach oben, unten, rechts oder links.

All diese Funktionen werden im Programm durch Symbole dargestellt. Diese lassen sich im CLI schlecht zuordnen, also: ausprobieren.

5 IconMake

5.1 Der Aufbau der Info-Dateien

Die modernen Benutzeroberflächen, wie z. B. *MS Windows*, *GEM* oder *Intuition*, sehen sich alle sehr ähnlich. Bei allen wird mit Hilfe einer sogenannten *»Maus«* ein kleiner Pfeil über den Bildschirm gelenkt, mit dem man auf kleine Bilder zeigen und sie *»anklicken«*, das heißt mit einem Druck auf eine Maustaste aktivieren kann. Diese kleinen Grafiksymbole nennt man *»Icons«*.
Diese Art der Kommunikation mit einem Computer ist sehr schnell erlernbar und erfordert keine speziellen Kenntnisse über den jeweiligen Computertyp oder dessen Betriebssystem. Man muß eigentlich nur noch wissen, wo sich der Netzschalter befindet, damit man den Computer ein- und ausschalten kann.
Wie fast jede Sache hat auch diese einen Haken, denn man erkauft sich diese Benutzerfreundlichkeit mit einem zum Teil sehr großen Verlust an Speicherkapazität. Das Betriebssystem benötigt schließlich eine Menge Informationen, um überhaupt alles so darstellen zu können, wie der Benutzer es auf dem Bildschirm sieht. So müssen zum Beispiel das Icon, das zu einem Programm gehört, sowie Informationen über eventuell zu öffnende Fenster oder vorher zu startende Programme irgendwo gespeichert sein.
Im folgenden Artikel wollen wir Ihnen detailliert zeigen, wie beim Amiga solch ein Icon aufgebaut ist. Mit diesen Informationen ausgestattet, wollen wir anschließend neue Darstellungsmöglichkeiten vorstellen.
Der Amiga verfügt über zwei verschiedene Benutzeroberflächen. Zum einen gibt es die Workbench mit ihren Icons, Fenstern und der bequemen Maussteuerung. Zum anderen existiert aber noch das sogenannte CLI, der *»Command Line Interpreter«*. Das CLI läßt die Maus völlig *»links liegen«* und akzeptiert nur Kommandos, die über die Tastatur eingegeben werden. Es hat aber einen großen Vorteil, denn es ermöglicht dem Benutzer, wirklich alle Dateien aufzulisten, die auf einer Diskette vorhanden sind.

So fällt es bei seiner Benutzung des <*Dir*>- oder <*List*>-Befehls sofort auf, daß es viele Dateien gibt, zu denen offensichtlich noch eine zweite gehört. Sie hat den gleichen Namen, nur ergänzt um die Endung »*.info*«. In diesen Dateien findet die Workbench alle notwendigen Informationen, um das Programm, die Schublade oder auch den Mülleimer als Icon auf dem Bildschirm darzustellen und um richtig auf ein Anklicken reagieren zu können.
Wie nun sind diese Dateien aufgebaut, und welche Informationen enthalten sie?
Eine schematische Übersicht über den Aufbau dieser Dateien liefert Bild 3. Wie man erkennen kann, ist die Datei in einzelne Blöcke und einige zusätzliche Daten unterteilt, deren Bedeutung im folgenden näher erläutert werden soll.
Vorher aber noch einige allgemeine Bemerkungen zu Bild 1. Die Einträge in den einzelnen Blöcken bestehen aus einer Zahl, die angibt, aus wie vielen Bytes der Eintrag besteht, aus einer kurzen Erläuterung und einer Zahl. Diese letzte Zahl (sie wird immer im Hexadezimalsystem angegeben) ist eine Konstante, die an dieser Stelle stehen muß, das heißt, man darf den Wert auf keinen Fall ändern, da das System sonst leicht abstürzen könnte. Falls keine weiteren Erläuterungen angegeben sind, handelt es sich um Einträge, die in diesem Zusammenhang unwichtig sind und auch keinen Einfluß auf die Icons haben.

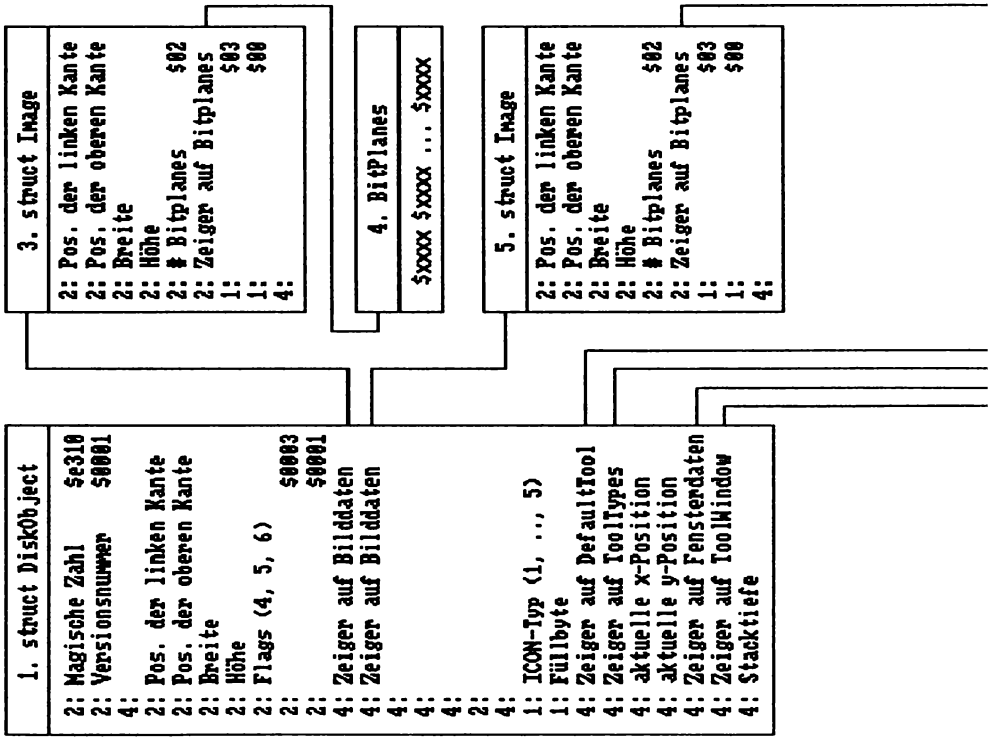


Bild 3. Der Aufbau einer ».info«-Datei, Teil 1

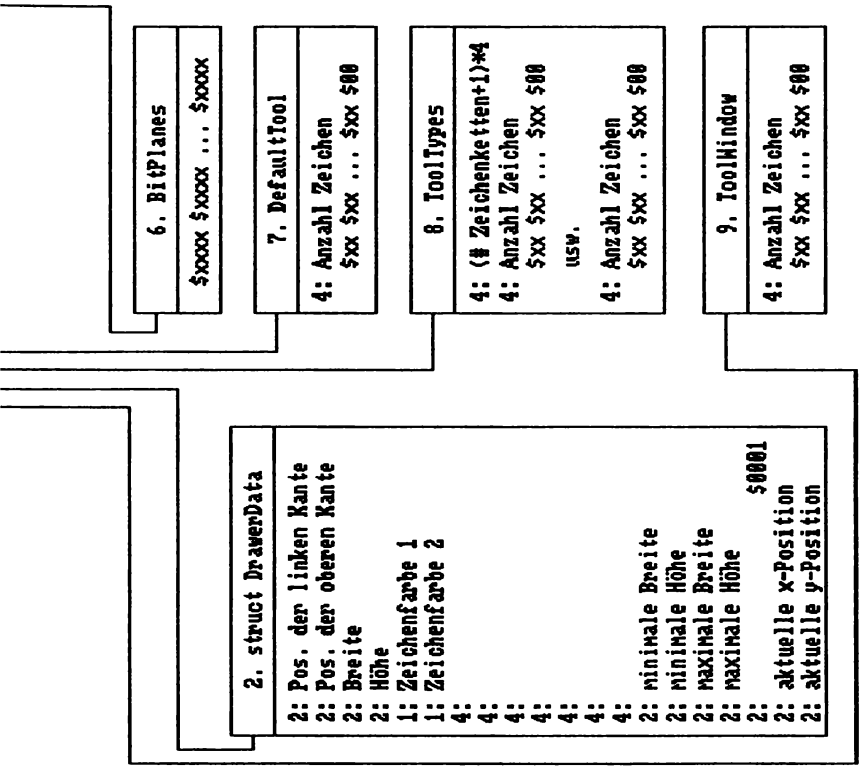


Bild 3. Der Aufbau einer ».info«-Datei, Teil 2

Kommen wir nun aber zur genauen Erläuterung der einzelnen Datenblöcke. Zunächst zwei Beispiele für den Dateiaufbau:

Beispiel 1: Eine mit *»Notepad«* erstellte Datei

DiskObject

E310 0001	Magische Nr./Version
00000000	
0061 000C 0028 0015	Position/Breite/Höhe
0004	Flag
0003 0001	
00033D8A 00000000	Zeiger auf Bilddaten
00000000 00000000 00000000	
0000 00000000	
04	ICON-Typ 4 => DateiICON
00	Füllbyte
00030580 00030570	DefaultTool/ToolTypes
80000000 80000000	akt. Position
00000000 00000000	Fensterdaten/ToolWindow
00000000	Stacktiefe

Image

0000 0000 0028 0015 0002	Position/Breite/Höhe/Tiefe
00033C8E	Zeiger auf Bitplane 1
03 00 00000000	

BitPlanes

0000 0000 0000 3FFF FFCC 0000 3FFF FFCF 0000	Bitplane 1
3FFF FFCF C000 3803 FFCF F000 3FFF FFC0 0000	
3803 FFFF FC00 3FFF FFFF FC00 3FFF FFFF FC00	
3F84 00C0 7C00 3FFF FFFF FC00 3900 8000 7C00	
3FFF FFFF FC00 3800 0040 7C00 3FFF FFFF FC00	

```
3FFF FFFF FC00 3FFF FE00 7C00 3FFF FFFF FC00
3FFF FFFF FC00 0000 0000 0000 0000 0000 0000

FFFF FFFC 0000 C000 0033 0000 C000 0030 C000  Bitplane 2
C000 0030 3000 C7FC 0030 0C00 C000 003F FF00
C7FC 0000 0300 C000 0000 0300 C000 0000 0300
C07B FF3F 8300 C000 0000 0300 C6FF 7FFF 8300
C000 0000 0300 C7FF FFBF 8300 C000 0000 0300
C000 0000 0300 C000 01FF 8300 C000 0000 0300
C000 0000 0300 FFFF FFFF FF00 0000 0000 0000
```

DefaultTool

```
00000012                                  Länge der Zeichenkette
55 54 49 4c 49 54 49 45 53 3A             Zeichenkette mit abschließender NULL
U  T  I  L  I  T  I  E  S  :
4E 6F 74 65 70 61 64 00
N  o  t  e  p  a  d
```

ToolTypes

```
00000010                                  # ToolTypes = (Zahl/4)–1
00000011                                  Länge der Zeichenkette
46 49 4C 45 54 59 50 45 3D 6E 6F 74 65 70 61 64 00
F  I  L  E  T  Y  P  E  =  n  o  t  e  p  a  d
0000000E                                  Länge der Zeichenkette
46 4F 4E 54 3D 74 6F 70 61 7A 2E 20 38 00
F  O  N  T  =  t  o  p  a  z  .     8
00000017                                  Länge der Zeichenkette
57 49 4E 44 4F 57 3D 20 20 30 2C 20 20 30 2C
W  I  N  D  O  W  =       0  ,        0  ,
                   36 34 30 2C 31 38 35 00
                   6  4  0  ,  1  8  5
```

Beispiel 2: Datei »*disk.info*« der Workbench 1.2

DiskObject

e310 0001	Magische Nr./Version
00000000	
01e3 0006 0020 0010	Position/Breite/Höhe
0005	Flag
0003 0001	
L0000c210 00000000	Zeiger auf Bilddaten
00000000 00000000 00000000	
0000 00000000	
01	ICON-Type 1 => DiskICON
74	Füllbyte
0001ad58 00000000	DefaultTool/ToolTypes
000001e3 00000006	akt. Position
0001b2b0 00000000	Fensterdaten/ToolWindow
00000000	Stacktiefe

DrawerData

0000 000b 0143 0063	Position/Breite/Höhe
ff ff	Zeichenfarben
00000000 0240027f 0001f02c 00000000	
0001e8d0 00000000 00000000	
005a 0028	minimale Breite/Höhe
ffff ffff	maximale Breite/Höhe
0001	
00000000 00000000	akt. Position d. Fensters

Image

0000 0000 0020 0010 0002	Position/Breite/Höhe/Tiefe
0001b470	Zeiger auf Bitplane 1
03 00 00000000	

BitPlanes

```
0000 0000 3ffc 3f00 3ffc 3fc0 3ffc 3ff0 BitPlane 1
3ffc 3ffc 3fff fffc 3fff fffc 3fff fffc
3fff fffc 3fff fffc 3fff fffc 3fff fffc
3fff fffc 3fff fffc 3fff fffc 0000 0000

ffff ffc0 c0ff fcf0 c0ff fc3c c0ff fc0f BitPlane 2
c0ff fc03 c0ff fc03 c000 0003 c000 0003
c000 0003 c000 0003 c000 0003 c000 0003
c000 0003 c000 0003 c000 0003 ffff ffff
```

DefaultTool

```
0000 0014                                Länge der Zeichenkette
53 59 53 3a 53 79 73 74 65 6d 2f         Zeichenkette mit abschließender NULL
S  Y  S  :  S  y  s  t  e  m  /
44 69 73 6b 43 6f 70 79 00
D  i  s  k  C  o  p  y
```

Nr. 1 Der »*DiskObject*«-Block

Die »*Magische Zahl*« sagt dem Betriebssystem, daß es sich bei dieser Datei um eine Info-Datei handelt. Die darauffolgende »*Versionsnummer*« für diesen Dateiaufbau ist bisher immer $0001. Weiterhin unterscheidet das Betriebssystem bei einem Icon zwei wesentliche Dinge: Das eigentliche Bild und den Bereich, in dem es ein Anklicken bemerken kann. Diese beiden Bereiche müssen nicht unbedingt übereinstimmen (siehe Bild 4). Die hier aufgeführte Position ist die des anklickbaren Bereiches. Es handelt sich dabei allerdings nicht um eine absolute Positionsangabe, sondern um ein Offset zu der in diesem Block etwas später angegebenen aktuellen Position des Icons.
Die Breite und die Höhe geben die Größe dieses unsichtbaren Bereiches an. Die Größen sind dabei übrigens immer in Pixeln dargestellt.

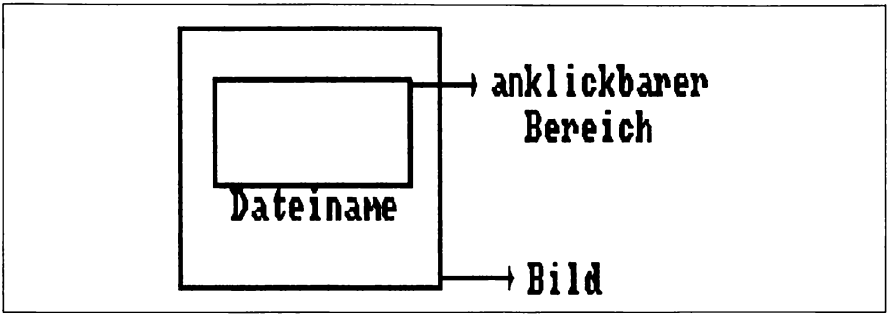


Bild 4. Das Icon-Bild und sein anklickbarer Bereich

»*Flag*« gibt an, was optisch passieren soll, wenn das Icon angeklickt, das heißt aktiviert wird. Es gibt hier drei Möglichkeiten:

1. Das gesamte Icon, das heißt das ganze rechteckige Feld, wird invertiert.
2. Das Bild wird invertiert, das heißt, nur die Punkte, die eine andere Farbe als die Hintergrundfarbe haben, werden anders dargestellt.
3. Es wird ein komplett neues Icon, das heißt ein Alternativ-Bild dargestellt. Diese Möglichkeit wird vom Icon-Editor allerdings nicht unterstützt.
   Je nach gewünschter Methode muß hier eine 4, 5 oder 6 eingetragen werden.

Die »*Zeiger auf die Bilddaten*« sind auf einen Block vom Typ »*IMAGE*« gerichtet, den wir gleich noch näher untersuchen werden. Der zweite Zeiger wird dabei nur benötigt, wenn »*Flag*« auf 6 gesetzt wurde.

Weiter geht es mit der Unterscheidung der fünf Icon-Typen, die es auf der Workbench gibt:

1. Das Disketten-Icon
2. Das Schubladen-Icon
3. Das Programm-Icon

4. Das Datei-Icon
5. Das Mülleimer-Icon

Je nach Icon-Typ ist der ganze Aufbau der Info-Datei etwas anders. Man kann also nicht einfach den Typ eines Icons ändern, ohne gleichzeitig alle für diesen Typ notwendigen Informationen bereitzustellen.
Nach dem Icon-Typ kommt immer ein Füllbyte, dessen Inhalt absolut unwichtig ist. Es wird nur benötigt, damit der Prozessor beim nächsten Zugriff wieder auf eine gerade Adresse zugreifen kann.
Das *»DefaultTool«* ist eine Zeichenkette, deren Bedeutung vom Icon-Typ abhängt und nur bei den Icon-Typen 1 und 4 vorkommt. Bei einem Datei-Icon gibt das *»DefaultTool«* an, welches Programm geladen werden muß, damit die in der Datei vorhandenen Daten richtig interpretiert werden. Jedes mit Amiga-Basic geschriebene Programm besitzt zum Beispiel ein Icon, dessen *»DefaultTool«* folgendes enthält: *»:AmigaBasic«*.
Will man nun ein Basic-Programm starten, dann muß man nur das Datei-Icon des Basic-Programms zweimal kurz nacheinander anklicken. Der Rest wird vom Betriebssystem erledigt. Das kann nämlich mit Hilfe des *»DefaultTools«* feststellen, daß es zuerst das Amiga-Basic laden muß, bevor das Basic-Programm laufen kann (siehe auch Beispiel 1). Bei dem Disketten-Icon sollte zum Beispiel immer folgendes stehen:

SYS:SYSTEM/DiskCopy

Dies gewährleistet, daß das Aufeinanderlegen von zwei Disketten-Icons auch zu dem im Handbuch angegebenen Ziel führt, nämlich das Kopieren der einen Diskette auf die andere (siehe Beispiel 2).
Die *»ToolTypes«* sind eine Reihe von Zeichenketten, die nur beim Icon-Typ 4 wichtig sind. Sie enthalten Zusatzinformationen für das zugehörige Programm (siehe wieder Beispiel 1).
Nun kommt die Position des Icons relativ zu dem Fenster, in dem es erscheint. Die Positionsangabe $80000000, $80000000 hat dabei eine besondere Bedeutung. Wird sie vorgefunden, dann sucht das Betriebssystem selbst einen freien Platz für das Icon. Sollte in dem sichtbaren Bereich des Fensters kein Platz mehr sein, positioniert das System das Icon in dem nicht sichtbaren Bereich, und man muß das Fenster vergrößern, um das Icon sehen zu können.

Der *»Zeiger auf Fensterdaten«* zeigt auf den *»DrawerData«*-Block, den wir gleich näher erläutern werden. Der *»Zeiger auf ToolWindow«* wird ebenfalls weiter unten erläutert.
Die *»Stacktiefe«* gibt, wie schon der Name sagt, die für das jeweilige Programm benötigte Tiefe des Stacks an. Ein Stack ist ein spezieller Speicherbereich, in dem während des Programmablaufes Daten, wie zum Beispiel Rücksprungadressen, abgelegt werden. Bei einer Stacktiefe von null wird automatisch der Wert 4096 eingesetzt. Beim Anklicken von Datei-Icons wird dessen Stacktiefe genommen und nicht die des vorher geladenen *»DefaultTools«*.

Nr. 2 Der *»DrawerData«*-Block

In diesem Datensegment sind alle Informationen zusammengefaßt, die für ein eventuell zu öffnendes Fenster unentbehrlich sind. Diese Daten werden bei den Disketten-Icons, den Schubladen und dem Mülleimer-Icon benötigt.
Zuerst kommt die Position des Fensters. Damit ist nicht die Position des Rahmens für den sichtbaren Teil des Fensters gemeint, sondern ein Offset zu dieser Rahmenposition. Danach ist Platz für die Zeichenfarben, mit denen der Rahmen und die sogenannten Gadgets gemalt werden.
Die Fenstergröße kann durch Angabe von *»minimum«* und *»maximum«* beschränkt werden. Den Schluß bildet die aktuelle Position des Fensterrahmens.

Nr. 3 + 5 Die *»Image«*-Blöcke

Hier findet man alle nötigen Informationen für die Darstellung des Icon-Bildes.
Zuerst kommt ein Offset zur aktuellen Icon-Position aus Block 1, gefolgt von der Breite und der Höhe des Bildes. Dann folgt die Anzahl der für das Bild nötigen BitPlanes. Da die Workbench normalerweise vierfarbig ist, benötigt man in der Regel zwei BitPlanes.
Als nächstes kommt der Zeiger auf die BitPlanes, die das eigentliche Bild enthalten. Die restlichen Bytes sollten nicht geändert werden.

Nr. 4 + 6 Die »*BitPlanes*«

Eine BitPlane enthält die Grafikdaten für das Bild. Die Anzahl der dafür benötigten Bytes berechnet man mit Hilfe der im »*Image*«-Block angegebenen Breite und Höhe nach folgender Formel:

*Anzahl Bytes = Höhe * 2 * INT ((Breite + 15) / 16)*

Das Betriebssystem liest die Grafikdaten immer wortweise, das heißt, 2 Byte auf einmal ein. Darum muß die Anzahl der Bytes für eine Grafikzeile immer gerade sein. Ist das Bild zum Beispiel 27 Pixel breit, dann werden immer 4 Byte pro Zeile gelesen.
Da hier die Anzahl der BitPlanes üblicherweise 2 ist, kommen immer (2 * Anzahl Bytes) Grafikdaten hintereinander.

Nr. 7 Das »*DefaultTool*«

Hinter diesem Begriff verbirgt sich, wie oben schon erklärt, eine Zeichenkette, die immer wie folgt aufgebaut ist:
Die ersten 4 Byte enthalten die Länge der Zeichenkette. Dann folgen die einzelnen Zeichen und zum Schluß noch eine Null ($00), die das Ende der Zeichenkette markiert.

Nr. 8 Die »*ToolTypes*«

Bei diesen Daten handelt es sich um eine ganze Reihe von Zeichenketten. Damit der Computer weiß, wie viele Zeichenketten folgen, steht hier zuerst eine Zahl, die sich wie folgt berechnet:

*4 * (Anzahl Zeichenketten + 1)*

Danach kommen die einzelnen Zeichenketten, wie unter Punkt 7 beschrieben.
Bei den »*ToolTypes*« handelt es sich um Zusatzinformationen für das zu startende Programm. Im Beispiel 1 finden wir als ersten Eintrag »*FILETYP = notepad*«. Mit Hilfe dieser Zeile können zum Beispiel Textverarbeitungsprogramme erkennen, daß es sich um eine Notepad-Datei handelt, und entsprechend reagieren.

Nr. 9 Das »*ToolWindow*«

Das »*ToolWindow*« ist ebenfalls eine Zeichenkette. Sie enthält Informationen für ein eventuell automatisch zu öffnendes Fenster und könnte zum Beispiel wie folgt aussehen:

con:0/0/640/200/Test

Diese recht praktische Möglichkeit wird aber momentan von fast keinem Programm unterstützt.
Die Nummern der Blöcke und der Zeichenketten geben die Reihenfolge an, in der sie gespeichert werden. Sie kommen zwar praktisch nie alle auf einmal vor, aber wenn sie vorhanden sind, dann liegen sie in der angegebenen Reihenfolge (siehe Beispiel 1 und Beispiel 2).
Will man nun irgend etwas an der Info-Datei ändern, dann sollte man dies nur mit Hilfe von speziellen Programmen, wie etwa dem im folgenden vorgestellten Programm »*IconMake*« tun, denn jeder kleine Fehler in dieser Datei kann sehr leicht zu einem Systemabsturz führen.
Zum Abschluß noch einige kleine Bemerkungen zu den vielfältigen Möglichkeiten, die dieser Dateiaufbau bietet. So ist es zum Beispiel kein Problem, ein Icon zu konstruieren, das den ganzen Bildschirm ausfüllt, denn der Block »*IMAGE*« erlaubt es, Bilder beliebiger Größe aufzunehmen. Ein Icon dieser Größe würde aber gleichzeitig alle anderen Icons verdrängen, da keine zwei Icons aufeinander liegen dürfen. Außerdem würde es sehr viel Speicherplatz kosten.
Es ist auch möglich, ein Icon zu basteln, bei dem der Dateiname von einem Rahmen umgeben ist (siehe als Beispiel das Icon zum »*IconMake*«). Der Name des Programmes, zu dem das Icon gehört, steht nämlich immer direkt unter dem Bereich, in dem das Betriebssystem ein Anklicken registriert. Zeichnet man nun ein Bild, das größer ist als dieser Bereich, dann erscheint der Dateiname mitten im Bild.
Durch die Möglichkeit, ein zweites Bild angeben zu können, kann man natürlich sehr schöne Effekte erzielen. So zum Beispiel sich öffnende Schubladen oder blitzende Fotoapparate.

5.2 IconMake

Was wir inzwischen über den Aufbau der Info-Dateien gelernt haben, zeigt uns, daß es nicht ohne weiteres möglich ist, hier Manipulationen vorzunehmen. Es sind einfach zu viele Dinge zu beachten, und zu schnell haben sich Fehler eingeschlichen. Da wir aber über einen Computer verfügen, liegt es nahe, ein Programm zu schreiben, das uns diese lästige Verwaltungsarbeit abnimmt. So entstand der *»IconMake«*.
Wie viele andere Programme auch, war der *»IconMake«* am Anfang als kleines Utility-Programm geplant. Doch im Gegensatz zu vielen anderen Programmen sollte Benutzerfreundlichkeit so groß wie irgend möglich werden, und so wurde das Programm größer und größer. In der vorliegenden Version bietet es nun allerdings ein (vorläufiges) Maximum an Komfort. Außerdem stellt es einige zusätzliche Möglichkeiten, wie etwa das Konvertieren von DPaint-Brushes in Icons und umgekehrt, zur Verfügung. Im folgenden wollen wir die verschiedenen Funktionen des *»IconMake«* näher erläutern und anschließend noch einige wichtige Tips und Hinweise für dessen Benutzung geben.
Nach dem Start des Programms öffnet sich ein Fenster, das den ganzen Bildschirm einnimmt und in dessen Mitte das Icon des *»IconMake«* erscheint. Nun stehen fünf Menüpunkte zur Verfügung, um alle Möglichkeiten der Info-Dateien auszuschöpfen und um die zusätzlichen Funktionen bereitzustellen. Doch kommen wir gleich zu den einzelnen Menüpunkten:

5.2.1 Das *»Project«*-Menü

Dieses erste Menü dient hauptsächlich dazu, fertige Icons zu laden oder zu speichern. Nach der Wahl von *»Load Icon«* erscheint ein *»Filename-Requester«*, dessen Funktionsweise weiter unten ausführlich erklärt wird. Hat man dort einen Namen angegeben, wird das Icon komplett geladen und im Arbeitsfenster angezeigt.
Bei *»Save Icon«* wird das konstruierte neue Icon gespeichert und kann jetzt direkt auf der Workbench getestet werden. Sollten beim Einlesen oder beim Speichern irgendwelche Schwierigkeiten auftreten, dann erscheint ein Requester mit einem (hoffentlich) erklärenden Hinweistext.
»Flip Icon« dient dazu, das Alternativbild, das heißt das Bild, das erscheint, wenn ein Icon aktiviert wurde, nach vorne zu holen. Das ursprünglich erste Bild wird dadurch zum neuen Alternativbild.

Kein Programm ohne Copyright-Erklärung. Deshalb auch hier ein eigener Menüpunkt für diese wichtige Meldung.
Verlassen kann man das Programm nur über *»Quit«*. Es erscheint zur Sicherheit noch ein Requester, der abfragt, ob man das Programm wirklich beenden will.

5.2.2 Das *»Picture 1«*-Menü

Hier wird die Sache schon interessanter, denn hier haben wir die Möglichkeit, DPaint-Brushes direkt einzulesen und darzustellen. Dies geschieht über das Menü *»Load«*. Wieder erscheint der Filename-Requester, und Sie werden gebeten, eine Datei auszuwählen. Das Programm erkennt nun von allein, ob Sie eine Info-Datei angegeben haben oder ob es sich um einen DPaint-Brush handelt.
Ist es eine Info-Datei, dann wird das erste (das vordere) Bild dieses Icons geladen. Ist es ein DPaint-Brush, dann werden die dort vorgefundenen Bilddaten konvertiert und das Bild dargestellt. Eine Beschränkung in der Größe des Bildes gibt es dabei theoretisch nicht. Man sollte nur daran denken, daß große Bilder auch sehr viel Speicherplatz belegen. So belegt zum Beispiel die Info-Datei für den *»IconMake«* schon fast 10 Kbyte im Speicher. Wie bei der Erläuterung des Aufbaus der Info-Dateien schon geschrieben, ist es auch kein Problem, ein DPaint-Brush zu laden, das so groß ist wie der ganze Bildschirm. Solche Icons können dem Betriebssystem aber einiges an Kopfzerbrechen bereiten, denn wo soll es auftauchende Icons unterbringen, wenn einfach kein Platz mehr da ist.
Jedes einzelne Bild kann auch wieder gespeichert werden, und zwar als Icon mit nur einem Bild oder als DPaint-Brush. Wird es als Icon gespeichert, bekommt es alle eingestellten Werte mit. Es unterscheidet sich von einem unter dem *»Project«*-Menü gespeicherten Icon nur dadurch, daß es kein Alternativ-Bild hat.
Die Möglichkeit, das Bild auch als DPaint-Brush zu speichern, erlaubt es, bereits vorhandene Icons mit Hilfe von DPaint an die eigenen Wünsche anzupassen. Auf den *»Fish-Disks«* findet man ganze Sammlungen mit fertigen Icons, die nur darauf warten, auf die eigene Workbench gebracht zu werden.

5.2.3 Das »*Picture 2*«-Menü

Dieses dritte Menü ist genauso aufgebaut wie das vorhergehende. Nur beziehen sich alle Angaben auf das Alternativbild des zu konstruierenden Icons.
Beim Laden eines Icons gibt es allerdings einen kleinen Unterschied. Besitzt das zu ladende Icon ein Alternativbild, wird dieses geladen. Ist keins vorhanden, wird automatisch das vordere Bild genommen (das ist eigentlich klar, denn was sollte man sonst nehmen?).

5.2.4 Das »*Icon Type*«-Menü

Hier kann man wählen, welchen Typ das fertige Icon haben soll. Um mehr braucht man sich nicht zu kümmern, denn die restliche Verwaltungsarbeit übernimmt das Programm.

5.2.5 Das »*Other*«-Menü

Wie immer kommt das Beste zum Schluß. Dieses letzte Menü bietet dem Benutzer nämlich die angekündigten Manipulations- und Einstellmöglichkeiten.
Als erstes kommen die diversen Tools, die bei der Erläuterung des Aufbaus der Icon-Datei schon näher erklärt wurden. Hier kann man ein eventuell benötigtes »*DefaultTool*« eingeben oder auch das bisher nicht unterstützte »*ToolWindow*«. Außerdem sind noch sechs Felder für eventuell schon vorher bekannte »*ToolTypes*« vorhanden.
Wieder muß man sich keine Gedanken darüber machen, ob man vielleicht etwas zu viel eingetragen hat oder nicht. Es werden immer nur die Dinge gespeichert, die für den jeweils ausgewählten Icon-Typ wichtig sind. Alle anderen Angaben werden ignoriert.
Der umfangreichste Requester ist der Parameter-Requester. Hier können all die Parameter eingestellt werden, die man bei einem Icon ändern kann, ohne Probleme mit dem Betriebssystem zu bekommen. Die eingegebenen Werte werden dabei zusätzlich in ihrem Wertebereich überwacht, so daß keine unsinnigen Zahlen eingegeben werden können. Versucht man es trotzdem, dann blitzt der Bildschirm kurz auf, und das Feld, in dem der falsche Wert eingegeben wurde, wird mit einem Standardwert gefüllt.
In der ersten Spalte können all die Werte verändert werden, die zu einem eventuell zu öffnenden Fenster gehören. Fenster werden, wie oben schon erwähnt, immer dann geöffnet, wenn man

ein Disketten-Icon (*disk*), ein Schubladen-Icon (*drawer*) oder aber das Mülleimer-Icon (*garbage*) aktiviert.
Die X- und die Y-Koordinaten sind die Koordinaten der linken oberen Ecke des Fensters. Außerdem kann man die Breite und die Höhe angeben. Mit *»Pen 1«* und *»Pen 2«* können die Farben eingestellt werden, mit denen der Fensterrahmen und die dazugehörigen Gadgets gezeichnet werden. Erlaubte Werte sind die Zahlen von 0 bis 3 und 255. Trägt man 255 ein, dann werden die Standardfarben der Workbench genommen.
Die zweite Spalte ist für die Icon-Werte. Die Koordinaten beziehen sich – wie beim Fenster – auf die linke obere Ecke des Icons. Allerdings gibt es hier eine kleine Besonderheit. Trägt man nämlich eine *4711* ein, dann sucht sich das Betriebssystem später selbst einen Platz für das Icon. Eigentlich hätte man, wie wir ja wissen, $80000000 eingeben müssen, aber das erschien mir zu lang, denn ins Zehnersystem umgerechnet ergibt das 2147483648, und wer kann sich diese Zahl schon merken.
Mit der Breite und der Höhe kann der anklickbare Bereich des Icons verändert werden. Besonders interessant ist hier die Höhe, denn damit kann die Position des Dateinamens beeinflußt werden, der, wie oben beschrieben, immer direkt unter dem anklickbaren Bereich steht. Als Beispiel für ein so manipuliertes Icon kann das *»IconMake«*-Icon dienen, dessen Namen innerhalb eines Rahmens steht.
Als letztes gibt es hier noch ein Feld, in dem die Stackgröße eingetragen werden kann. Läßt man es leer, wird die Standardgröße von 4096 Byte übernommen.
Will man die gemachten Änderungen übernehmen, muß man nur noch den Requester mit *»OKAY«* verlassen. Beim Speichern des Icons werden diese dann automatisch eingesetzt und mitgespeichert.
Der Menüpunkt *»Picture Data«* dient nur der Information. Er zeigt die Breite und die Höhe der beiden geladenen Bilder.
Nachdem die einzelnen Menüpunkte erklärt sind, möchte ich, wie angekündigt, zum Abschluß noch einige Tips und Hinweise geben.
Wer sich darüber ärgert, daß das Fenster den ganzen Bildschirm einnimmt und auch nicht in der Größe veränderbar ist, für den wurde der folgende kleine Gag eingebaut. Versucht man, das Programm mit Hilfe des Schließ-Gadgets in der linken oberen Ecke zu beenden, muß man feststellen, daß etwas ganz anderes passiert. Es erscheint nämlich ein kleines Icon, das, wie

es bei Icons üblich ist, mit Hilfe der Maus über den Bildschirm bewegt werden kann. Klickt man dieses Icon zweimal kurz hintereinander an, dann erscheint wieder das alte *»IconMake«*-Fenster.
Durch diese Methode ist es möglich, seine Icons zu testen, ohne das Programm verlassen zu müssen. Man kann es statt dessen *»in eine Ecke legen«*.
Damit das Programm auch optisch den richtigen Eindruck hinterläßt, sollte sich im Verzeichnis *FONTS:* der Zeichensatz *»garnet 9«* befinden. Das Programm funktioniert zwar auch ohne, aber die Optik leidet etwas.
Da es immer noch Leute gibt, die meinen, daß man mit der Tastatur schneller arbeiten kann als mit der Maus, können sämtliche Menüpunkte auch über die Tastatur aktiviert werden. Außerdem können die hier benutzen Requester wie gewohnt mit *<linke Amiga-Taste>* + *<v>* (für *»ok«*) und *<linke Amiga-Taste>* + *<b>* (für *»cancel«*) beendet werden.
Beim Laden von kompletten Icons werden die dort vorhandenen Tools und Parameter mit übernommen. Dies soll verhindern, daß ein Icon falsche Tools bekommt oder gar vom falschen Typ ist, wenn man es auf die Schnelle lädt, mit einem Alternativbild versieht und dann wieder speichert.
Das Beste ist, man macht es sich zur Gewohnheit, alle Einstellungen vor dem Speichern noch einmal zu kontrollieren. Zu schnell vergißt man es, den richtigen Typ einzustellen oder das *»DefaultTool«* anzugeben. Lädt man einzelne Bilder, dann wird nichts verändert, außer der Größe des anklickbaren Bereiches. Dieser wird automatisch auf die Größe des größten Bildes gebracht.
Will man ein komplett neues Icon konstruieren, arbeitet man am besten mit DPaint, um die beiden dazugehörigen Bilder zu malen. Dieses Programm stellt die meisten Hilfsmittel für ein vernünftiges Arbeiten zur Verfügung. Man sollte allerdings darauf achten, daß man auch die Workbench-Auflösung (640 x 256 bei 4 Farben) eingestellt hat. Nur dann ist sichergestellt, daß das Icon nachher richtig *»funktioniert«*.
Bei Icons mit Alternativbildern sollte man darauf achten, daß beide Bilder gleich groß sind. Es kann sonst zu unerwünschten Nebeneffekten kommen. Es hat sich dabei als sehr hilfreich erwiesen, die Koordinatenanzeige bei DPaint einzuschalten. Schneidet man nämlich ein Brush aus, geht diese Anzeige sofort auf Null und zeigt einem die aktuelle Höhe und Breite des auszuschneidenden Sektors an.

Da DPaint zwei komplette Bildschirmseiten zur Verfügung stellt, kann man das neue Icon in DPaint schon im voraus testen. Beide Bilder werden dazu natürlich zuerst erstellt und dann in je einem der beiden Seiten so plaziert, daß sie genau übereinander liegen. Mit der Taste <*j*> kann man nun (wie mit der Maus bei einem Icon) zwischen beiden Bildern hin und her wechseln.

5.2.6 Der Filename-Requester

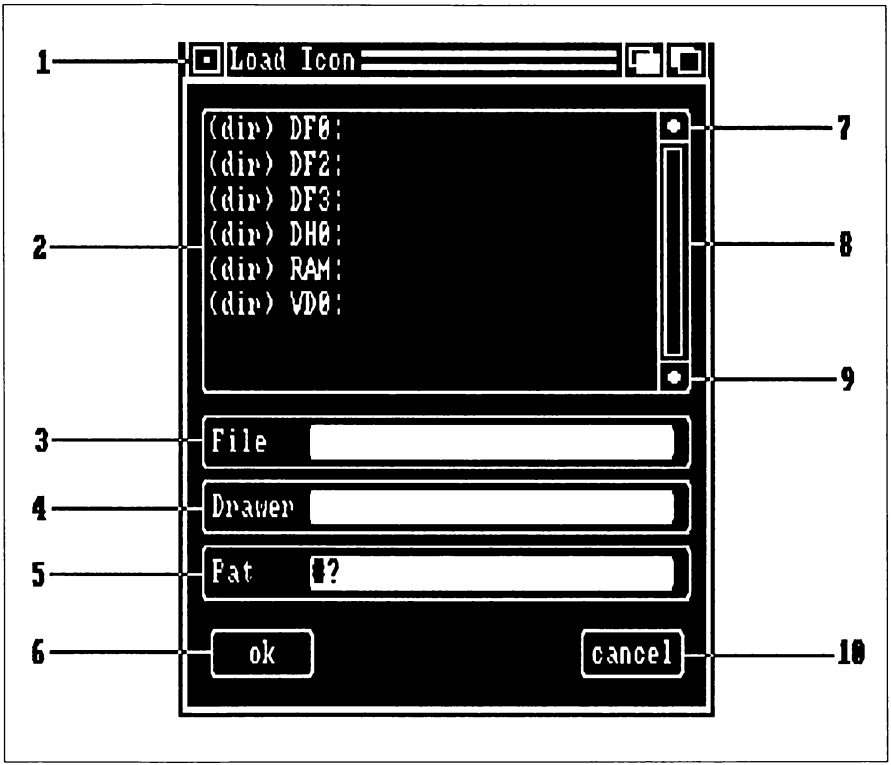


Bild 5. Der Filename-Requester

Jedes gute Amiga-Programm verfügt über einen mehr oder weniger komfortablen Filename-Requester (eine nicht besonders gute deutsche Übersetzung wäre: Dateinamen-Anforderungs-Fenster), einen Mechanismus also, der den Benutzer dazu auffordert, einen Dateinamen anzugeben.
Der hier benutzte Requester (siehe Bild 5) öffnet ein eigenes Fenster, das wie jedes andere Fenster auch verschoben, in den Hintergrund bzw. Vordergrund verlagert und geschlossen werden kann. In diesem Fenster gibt es nun verschiedene Felder, die im folgenden näher erklärt werden sollen.
Das wichtigste Feld ist das mit der Nummer *2*. Es enthält nämlich die Liste der in dem momentan aktuellen Verzeichnis vorhandenen Dateieinträge. Unterverzeichnisse werden dabei mit einem vorangestellten *»(dir)«* gekennzeichnet. Mit dem Schieberegler *(8)* kann man sich die komplette Liste anschauen. Ist die gewünschte Datei aufgetaucht, dann muß man sie nur anklicken. Sie erscheint dann sofort im Feld *3*. Falls man eine neue Datei anlegen will, kann man hier selbstverständlich auch einfach den neuen Namen eintragen.
Um eins der angezeigten Verzeichnisse zum aktuellen zu machen, braucht man den Namen ebenfalls nur anzuklicken. Sofort beginnt das Programm mit der Auflistung der dort vorgefundenen Dateien. Der Name des ausgewählten Verzeichnisses erscheint dabei in dem Feld mit dem Namen *»Drawer«* (*4*). Auch hier kann der Benutzer selbstverständlich per Hand ein anderes Verzeichnis eingeben.
Will man zurück in das übergeordnete Verzeichnis, dann genügt ein Mausklick auf das Feld mit der Nummer *9*. Bei jedem Anklicken steigt man dabei eine Stufe im Dateibaum höher. Ein Anklicken im Feld *7* bewirkt, daß man eine komplette Liste der angeschlossenen Laufwerke bekommt.
Für die Leute, die besonders viele Dateien in einem Verzeichnis haben, ist das Feld *5*. Der Name *»Pat«* steht nämlich für *»Pattern Matching«*. Hier kann man, wie beim CLI auch, ein Muster für die Auswahl der anzuzeigenden Dateien angeben. Will man sich zum Beispiel nur die Info-Dateien ansehen, dann muß hier nur folgendes eingetragen werden: »#?.info«. Die vorhandenen Verzeichnisse werden trotzdem komplett angezeigt.
Das Feld *»Cancel« (10)* und das Schließ-Gadget *(1)* bewirken einen Abbruch der Dateiauswahl. Bei *»ok«* dagegen wird der ausgewählte Dateiname mit der kompletten Pfadangabe der aufrufenden Routine zur Verfügung gestellt.

Der Requester ist dabei voll an das Multitasking des Amiga angepaßt. Jede Aktion, die möglich ist, kann auch zu jeder Zeit ausgeführt werden. Hat man sich zum Beispiel beim Aufruf vertan, dann kann man sofort *»Cancel«* anklicken, und der Requester verschwindet wieder, ohne daß man darauf warten muß, daß das gesamte Verzeichnis gelesen wurde, wie es etwa bei DPaint der Fall ist.

5.3 Icon_to_C

Das Programm *»Icon_to_C«* ist ein Hilfsprogramm für alle die Leute, die spezielle Icons in ihre eigenen Programme einbauen möchten.
Hat man zum Beispiel seinen eigenen kleinen Text- oder Grafikeditor geschrieben und möchte nun, daß die mit diesem Editor angelegten Dateien sofort an einem eigenen Icon erkannt werden, dann hat man zwei Möglichkeiten:

1. Man kopiert jedesmal ein bestehendes Muster-Icon.
2. Man baut das gewünschte Icon in den Object Code des Editors mit ein.

Die erste Methode ist recht einfach in der Handhabung und erfordert keine allzugroße Arbeit; sie hat allerdings einen wesentlichen Nachteil: wird nämlich aus Versehen das zu kopierende Muster gelöscht, dann funktioniert sie nicht mehr.
Die zweite Methode ist wesentlich aufwendiger und erfordert einiges Wissen über den Aufbau von Icons.
Wie eine Info-Datei auf der Diskette gespeichert ist und was die dort vorhandenen Blöcke bedeuten, konnte man in dem Artikel *»Der Aufbau der Info-Datei«* nachlesen. Will man das Icon aber in sein Programm einbauen, dann benötigt man diese Informationen im C-Source-Code.
Natürlich kann man alle diese Daten per Hand eingeben und dann hoffen, daß man sich nicht vertan hat. Viel einfacher wäre es aber, wenn man ein Icon mit dem *»IconEd«* oder dem *»Icon-Make«* komplett fertig machen und dann direkt in das benötigte Format umwandeln könnte.
Hier kommt nun das Programm *»Icon_to_C«* ins Spiel. Es nimmt sich nämlich ein fertiges Icon, liest die dort vorhandenen Daten ein und wandelt es in ein C-Programm um. Dieses C-Programm muß dann nur noch mit Hilfe eines Compilers übersetzt und an das Hauptprogramm angebunden werden.

Da das Umwandlungsprogramm außerdem Kommentare in den Source-Code einfügt, können kleine Änderungen an dem jeweiligen Icon leicht vorgenommen werden.

Aufgerufen wird das Programm wie folgt:

Icon_to_C <Name der Info-Datei ohne *».info«>*

Als Ergebnis entsteht dabei eine Datei mit dem Namen *<Name>.c,* die den Source-Code des Icons enthält.
Hat man zum Beispiel eine Notepad-Datei mit dem Namen *»Text«*, dessen Icon man umgewandelt haben möchte, dann gibt man ein:

Icon_to_C Text

Als Ergebnis erhält man dann die Datei *»Text.c«*, in der der gewünschte *»struct DiskObject Text«* zu finden ist.
Mit Hilfe des Aufrufs *»PutDiskObject(<Name>, &Text)«* kann nun vom Hauptprogramm aus eine Kopie des Icons unter dem Namen *<Name>.info* gespeichert werden.
Das Programm *»Icon_to_C«* ist möglichst übersichtlich aufgebaut und besteht fast nur aus Ausgabe-Anweisungen. Jeder Block des Icons wird in einer eigenen Unterroutine abgehandelt.

Bücher zum Amiga

M. Breuer
DELUXE-Grafik mit dem Amiga
1987, 370 Seiten
Eine behutsame Einführung in die grundlegenden Konzepte des jeweiligen Programms. Führt anhand kleiner überschaubarer Beispiele die wichtigsten Programmbefehle vor. Ein Nachschlageteil zu jedem Programm listet alle Befehle und ihre Bedeutung auf. Den Abschluß der Beschreibung jedes Programms bildet eine Sammlung von Tips und Tricks.
- Das deutsche Handbuch für den kreativen Grafikkünstler mit der DELUXE-Serie.

Bestell-Nr. 90412
ISBN 3-89090-412-2
DM 49,–
(sFr 45,10/öS 382,20)

Bestell-Nr. 90236
ISBN 3-89090-236-7
DM 49,–
(sFr 45,10/öS 382,20)

M. Kohlen
Grafik auf dem Amiga
1987, 337 Seiten
Dieses Buch enthält eine ausführliche Beschreibung der Grafik-Hard- und -Software, deren Funktionsweise und führt in die Grundzüge der Grafikprogrammierung ein. In den folgenden Kapiteln werden diese Kenntnisse dann in praktischen Beispielen umgesetzt. Außerdem bietet das Buch einen Überblick über die vorhandenen Soft- und Hardware-Erweiterungen für den Amiga.
- Eine Pflichtlektüre für jeden, der sich für die phantastische Grafik des Amiga interessiert.

Markt&Technik
Zeitschriften · Bücher
Software · Schulung

Markt&Technik-Produkte erhalten Sie bei Ihrem Buchhändler, in Computer-Fachgeschäften oder in den Fachabteilungen der Warenhäuser.

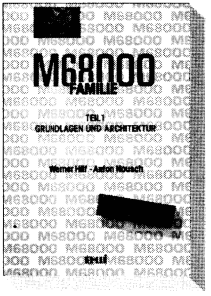
712425

Markt&Technik Verlag AG, Buchverlag, Hans-Pinsel-Straße 2, 8013 Haar bei München, Telefon (089) 4613-0

Bücher zum Amiga

Prof. D. A. Lien
Amiga: Programmier-Praxis mit MS-Basic
1986, 394 Seiten
Bestseller! Eine systematische und lebendige Einführung in MS-Basic unter der komfortablen Mausbedienung und Fensteroberfläche des Amiga. Mit über 60 Musterprogrammen zu den Befehlen. Zeigt Amiga-typische Anwendungen: bewegte/farbige Grafiken; Musik- und Sprachausgabe, Strings, Felder, Mathematik, Dateibehandlung, Ein-/Ausgabe sowie »Entwurf von Programmen«.

Bestell-Nr. 80369
ISBN 3-921803-69-1
DM 59,–
(sFr 54,30/öS 460,20)



Bestell-Nr. 80316
ISBN 3-921803-16-0
DM 79,–
(sFr 72,20/öS 616,20)

W. Hilf/A. Nausch
M68000-Familie: Teil 1 Grundlagen und Architektur
1984, 568 Seiten
Ausbildungs- und Entwicklungstext mit allen notwendigen Informationen über den M68000.

W. Hilf/A. Nausch
M68000-Familie: Teil 2 Anwendungen und 68000-Bausteine
1985, 400 Seiten
In vielen Programmierbeispielen liefert dieses Buch die Praxis der in Teil 1 vermittelten Theorie.
Bestell-Nr. 80330
ISBN 3-921803-30-6
DM 69,–
(sFr 63,50/öS 538,20)

Markt&Technik
Zeitschriften · Bücher
Software · Schulung

Markt&Technik-Produkte erhalten Sie bei Ihrem Buchhändler, in Computer-Fachgeschäften oder in den Fachabteilungen der Warenhäuser

712424

Markt&Technik Verlag AG, Buchverlag, Hans-Pinsel-Straße 2, 8013 Haar bei München, Telefon (089) 4613-0

Bücher zum Amiga

H.-R. Henning
Programmieren mit Amiga-Basic
1987, 363 Seiten,
inkl. Diskette
Einführung in die Programmierung des Amiga-Basic: Grafik, Sprites, Sprachausgabe, sequentielle Dateien, Fenstertechnik, Musik, Tips und Tricks. Hard- und Software-Anforderungen: Amiga 500, 1000 oder 2000 mit 512 Kbyte Arbeitsspeicher, gegebenenfalls ein grafikfähiger Matrixdrucker und ein Joystick, Amiga-Basic von Microsoft

Bestell-Nr. 90434,
ISBN 3-89090-434-3
DM 59,–
(sFr 54,30/öS 460,20)

Bestell-Nr. 90465
ISBN 3-89090-465-3
DM 59,–
(sFr 54,30/öS 460,20)

Commodore-Amiga Inc.
Das Amiga-DOS-Handbuch für Amiga 500, 1000 und 2000
1988, 342 Seiten
Die Pflichtlektüre für jeden Commodore-Amiga-Anwender und Programmierer: eine Entwickler-Dokumentation zum Amiga-DOS-Betriebssystem, Version 1.2. Programmierung, interne Datenstruktur und Diskettenhandling. Mit diesem Buch lernen Sie das mächtige Amiga-DOS schnell und sicher zu beherrschen. Alle Möglichkeiten des Systems, bis hin zum »Multi-Tasking« werden ausführlich und anschaulich beschrieben.

Markt&Technik
Zeitschriften · Bücher
Software · Schulung

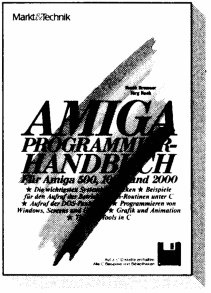
Markt&Technik-Produkte erhalten Sie bei Ihrem Buchhändler, in Computer-Fachgeschäften oder in den Fachabteilungen der Warenhäuser.

712423

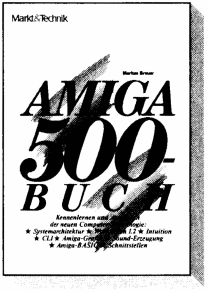
Markt&Technik Verlag AG, Buchverlag, Hans-Pinsel-Straße 2, 8013 Haar bei München, Telefon (089) 4613-0

Bücher zum Amiga

Kremser/Koch
Amiga-Programmierhandbuch
1987, 387 Seiten,
inkl. Diskette
Eine tolle Einführung in die »Interna« des Amiga: Die wichtigsten Systembibliotheken, die das Betriebssystem zur Verfügung stellt, werden anhand vieler Beispiele erklärt. Aus dem Inhalt: Aufruf der Betriebssystem-Routinen unter C, Aufruf der DOS-Funktionen, Programmieren von Windows, Screens und Gadgets, Grafik und Animation, Tips und Tools in C.



Bestell-Nr. 90491
ISBN 3-89090-491-2
DM 69,–
(sFr 63,50/öS 538,20)



Bestell-Nr. 90522
ISBN 3-89090-522-6
DM 49,–
(sFr 45,10/öS 382,20)

M. Breuer
Das Amiga-500-Handbuch
1987, 489 Seiten
Eine ausführliche Einführung in die Bedienung des Amiga 500. Kennenlernen und Anwenden der neuen Computer-Technologie: Systemarchitektur, Workbench 1.2, Intuition, CLI, Amiga-Grafik, Sound-Erzeugung, Amiga-Basic und Schnittstellen. Neben dem Handbuchteil mit vielen Bildschirmfotos und Übersichtstabellen, die Ihnen beim täglichen Einsatz helfen, schnell und reibungslos zu arbeiten, enthält das Buch eine ausführliche Beschreibung des Amiga 500 und seines Zubehörs.

Markt&Technik
Zeitschriften · Bücher
Software · Schulung

Markt&Technik-Produkte erhalten Sie bei Ihrem Buchhändler, in Computer-Fachgeschäften oder in den Fachabteilungen der Warenhäuser.

712422

Markt&Technik Verlag AG, Buchverlag, Hans-Pinsel-Straße 2, 8013 Haar bei München, Telefon (089) 4613-0

Computerliteratur und Software vom Spezialisten

Vom Einsteigerbuch für den Heim- oder Personalcomputer-Neuling über professionelle Programmierhandbücher bis hin zum Elektronikbuch bieten wir Ihnen interessante und topaktuelle Titel für

• Apple-Computer • Atari-Computer • Commodore 64/128/16/116/Plus 4 • Schneider-Computer • IBM-PC, XT und Kompatible

sowie zu den Fachbereichen Programmiersprachen • Betriebssysteme (CP/M, MS-DOS, Unix, Z80) • Textverarbeitung • Datenbanksysteme • Tabellenkalkulation • Integrierte Software • Mikroprozessoren • Schulungen.

Außerdem finden Sie professionelle Spitzen-Programme in unserem preiswerten Software-Angebot für Amiga, Atari ST, Commodore 128, 128 D, 64, 16, für Schneider-Computer und für IBM-PCs und Kompatible!

Fordern Sie mit dem nebenstehenden Coupon unser neuestes Gesamtverzeichnis und unsere Programmservice-Übersichten an, mit hilfreichen Utilities, professionellen Anwendungen oder packenden Computerspielen!

Markt&Technik

Zeitschriften · Bücher

Software · Schulung

Markt&Technik Verlag AG, Buchverlag, Hans-Pinsel-Straße 2, 8013 Haar bei München, Telefon (089) 4613-0

709006

Bitte schneiden Sie diesen Coupon aus, und schicken Sie ihn in einem Kuvert an: Markt&Technik Verlag AG, Buchverlag, Hans-Pinsel-Straße 2, 8013 Haar

Markt&Technik Verlag AG
– Unternehmensbereich Buchverlag –
Hans-Pinsel-Straße 2
D-8013 Haar bei München

Bitte schicken Sie mir:

☐ Ihr neuestes Gesamtverzeichnis

☐ Eine Übersicht Ihres Programmservice-Angebotes aus der Zeitschrift

_____

_____

☐ Außerdem interessiere ich mich für folgende/n Computer:

_____

_____

(PS.: Wir speichern Ihre Daten und verpflichten uns zur Einhaltung des Bundesdatenschutzgesetzes)

Adresse: Name _____ Straße _____ Ort _____

Raum für Notizen

Raum für Notizen

Raum für Notizen

AMIGA

Markt&Technik

SOFTWARE EXTRA

GRAFIK

Enthalten:
Eine 3½"-Programmdiskette

In seiner Grafikfähigkeit nimmt der Amiga unangefochten den Spitzenplatz in seiner Computerklasse ein. Glauben Sie etwa, Sie könnten keine aufwendige, schnelle Grafik in Ihre Programme einflechten, da Sie »nur« Amiga-Basic oder C beherrschen? Dann schauen Sie sich erst einmal die Programme auf dieser Diskette an.

- **Amiga Object Editor:** Spielerisch Bobs erzeugen, sei es über den integrierten Editor oder über IFF-Brushes. Animation inbegriffen.
- **The Bobbit:** IFF-Bilder in Bobs und Images verwandeln oder mit dem eingebauten Malprogramm erstellen.
- **Iconmake:** Generieren von animierten Icons und deren Wandel in C-Programme.
- **Booter:** Bootblock-Grafik par excellence.

Hardware-Anforderungen:
Amiga 500, 1000, 2000 mit mindestens 512 Kbyte Speicherkapazität

Lieferumfang:
Anleitungsheft
3½"-Diskette

Markt&Technik
Hans-Pinsel-Straße 2
D-8013 Haar bei München

Unverbindliche Preisempfehlung

DM 49,-
sFr 45,-
öS 490,-

4 001057 387256

This was brought to you

from the archives of

http://retro-commodore.eu